A
MST Practical Activity Report
Submitted for

ENGINEERING DESIGN-II (UTA024)

Submitted by:

102383026 Anil Kumar

102383024 Ananya Bera

102383027 Ansh Gandhi

102383028 Apurv Sharma

BE Second Year

Batch – 2CO15

Submitted to:

Dr. Ashutosh Mishra

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Computer Science and Engineering Department

TIET, Patiala

Jan-June 2024

ABSTRACT

These hands on experiments introduces the core concepts of Arduino programming through LED manipulation, serial communication, and numerical conversions. Participants start with basic tasks like blinking an LED, then progress to create intricate patterns across multiple LEDs. By manipulating LED intensity and exploring sequential patterns, they gain experience with both digital and analog control.

The experiment delves into converting decimal numbers to various representations, building a solid understanding of different numerical systems. Hands-on practice with for loops and serial communication functions further solidifies knowledge and prepares participants for advanced embedded systems development.

DECLARATION

We declare that this project report is based on our own work carried out during the course of our study in our Engineering-design II Computer Lab under the supervision of Mr. Ashutosh Mishra.

We assert that the statements made and conclusions drawn are an outcome of our own research work.

We further certify that the work contained in this report is original and has been done by us under the general supervision of our supervisor.

We have followed the guidelines provided by the University in writing this report.

We also declare that this project is the outcome of our own effort, that it has not been submitted to any other university for the award of any degree.

Date: February 20, 2024

TABLE OF CONTENTS

ABSTRACT	1
DECLARATION	2
TABLE OF CONTENTS	3
LIST OF TABLES	4
LIST OF FIGURES	5
INDEX	6

LIST OF TABLES

Table No.	Caption	Page No.
Table 1.1	Hardware Used	7
Table 2.1	Hardware Used	9
Table 3.1	Hardware Used	12
Table 4.1	Hardware Used	14
Table 5.1	Hardware Used	18
Table 6.1	Hardware Used	20
Table 7.1	Hardware Used	23
Table 8.1	Hardware Used	28
Table 9.1	Hardware Used	32

LIST OF FIGURES

Figure No.	Caption	Page No.
Figure 1.1	Arduino Uno R3	7
Figure 2.1	Resistors	9
Figure 2.2	LEDs	9
Figure 2.3	Arduino Uno R3	10
Figure 2.4	Breadboard	10
Figure 2.5	Jumper Wires	10
Figure 2.6	Blink a single LED using Arduino and breadboard	10
Figure 2.7	Blink a single LED using Arduino and breadboard	11
Figure 3.1	Various Resistors	12
Figure 3.2	LEDs	12
Figure 3.3	Arduino Uno R3	13
Figure 3.4	Breadboard	13
Figure 3.5	Jumper wires	13
Figure 3.6	Circuit for 5LEDs blinking in forward and reverse pattern	13
Figure 3.7	Circuit for 5LEDs blinking in forward and reverse pattern	14
Figure 4.1	Sequence of multiple LED’s	16
Figure 4.2	Sequence of multiple LED’s	17
Figure 5.1	Control brightness of LED	19
Figure 5.2	Control brightness of LED	19
Figure 6.1	Printed pattern using for loop	22
Figure 7.1	Circuit for Change intensity of multiple LEDs	24
Figure 7.2	Circuit for Change intensity of multiple LEDs	27
Figure 8.1	PWM Signal	28
Figure 8.2	Change the intensity of the given LED’s for the sequence	29
Figure 8.3	Change the intensity of the given LED’s for the sequence	31
Figure 9.1	Nvis 3302ARD Model	32

INDEX

S. No.	Name of Experiments	Page No.
1	Introduction to Arduino Micro-Controller.	7-8
2	To make a circuit of blinking an LED using Arduino Uno and breadboard.	9-11
3	To blink multiple LEDs using Arduino Uno and breadboard.	12-14
4	Write a program to design a pattern of sequence of multiple LED's using for loop in Arduino.	15-17
5	Write a program to demonstrate sending data from the computer to the Arduino board and control brightness of LED.	18-20
6	*Serial Communication:* WAP to print following pattern using for loop. *** Roll_No.____ *************************************** Name:____ *************************************** Branch:_______________________________________ ***************************************	21-22
7	WAP to show dimmer effect where LED 1 should display values between 1-50 LED 2 = 51-100 LED 3 = 101-150 LED 4 = 151-200 LED 5 = 201-255	23-27
8	Write a program to change the intensity of the given LED's for the sequence 35214 in for both forward and reverse order.	28-31
9	Write a program to demonstrate control of DC Motor using forward, backward, left, right turn motion and clock- wise/anti clock- wise rotation.	32-34

EXPERIMENT-1

OBJECTIVE: Introduction to Arduino Micro-Controller.

SOFTWARE USED: Arduino

HARDWARE USED:

Sr. No	Name of Components	Value
1	Arduino Uno Micro-Controller	1

Table 1.1: Hardware Used

LOGIC/CIRCUIT DIAGRAM:

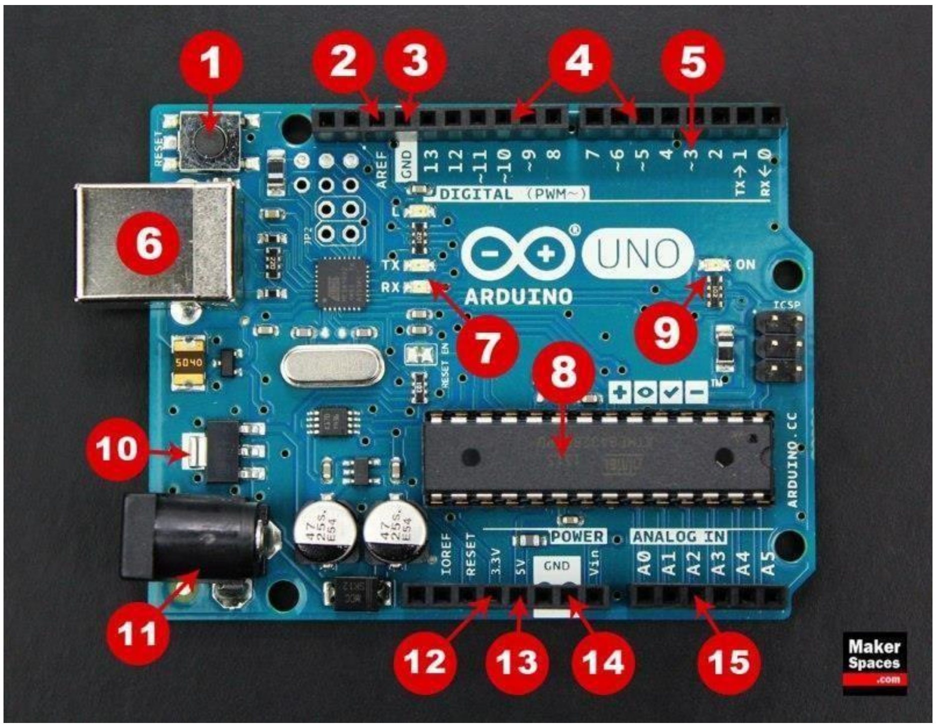


Fig 1.1: Arduino Uno R3

THEORY:

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc and initially released in 2010.

The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the USB cable or by an external 9-volt battery,though it accepts voltages between 7 and 20 volts.

1. **Reset Button** – This will restart any code that is loaded to the Arduino board.

2. **AREF** – Stands for "Analog Reference" and is used to set an external reference voltage.

3. **Ground Pin** – There are a few ground pins on the Arduino and they all work the same.

4. **Digital Input/Output** – Pins 0-13 can be used for digital input or output.

5. **PWM** – The pins marked with the (~) symbol can simulate analog output.

6. **USB Connection** – Used for powering up your Arduino and uploading sketches.

7. **TX/RX** – Transmit and receive data indication LEDs.

8. **ATmega Microcontroller** – This is the brains and is where the programs are stored

9. **Power LED Indicator** – This LED lights up anytime the board is plugged in a power source.

10. **Voltage Regulator** – This controls the amount of voltage going into the Arduino board.

11. **DC Power Barrel Jack** – This is used for powering your Arduino with a power supply.

12. **3.3V Pin** – This pin supplies 3.3 volts of power to your projects.

13. **5V Pin** – This pin supplies 5 volts of power to your projects.

14. **Ground Pins** – There are a few ground pins on the Arduino and they all work the same.

15. **Analog Pins** – These pins can read the signal from an analog sensor and convert it to digital.

DISCUSSION

In this experiment, we get to know about basics of Arduino Uno Microcontroller and its various functions and components.

Signature of faculty member

EXPERIMENT-2

OBJECTIVE: Write a program to blink a single LED using Arduino and breadboard.

SOFTWARE USED: Arduino

HARDWARE USED:

Sr No.	Name of the Component	Value
1.	Arduino Uno Board	1
2.	Breadboard	1
3.	Jumper Wires	2
4.	LED	1
5.	Resistor	220 ohm

Table 2.1: Hardware Used

THEORY:

Resistor: Resistors are used in virtually all electronic circuits and many electrical ones. Resistors, as their name indicates resist the flow of electricity and this function is key to the operation most circuits.

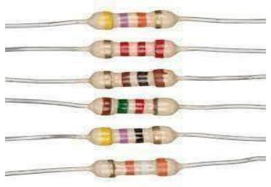


Fig 2.1 : Resistors

LED: A light-emitting diode (LED) is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons (Energy packets).

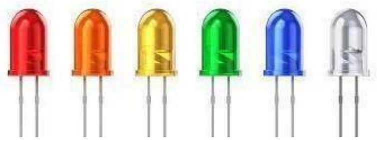


Fig 2.2 : LEDs

Arduino Uno Board: The **Arduino Uno** is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc.

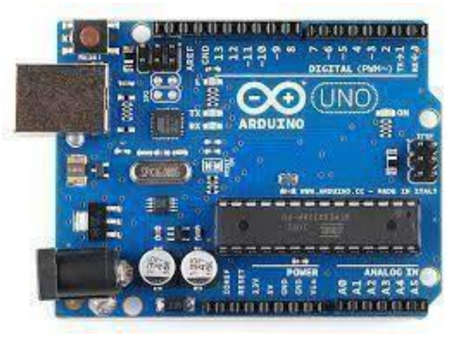


Fig 2.3 : Arduino Uno R3

Breadboard: A breadboard is used to place components (resistor, capacitor, LED's etc.) that are wired together. It is used to make temporary circuits.

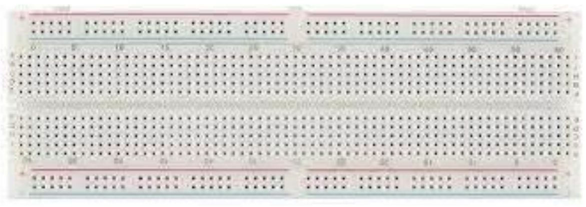


Fig 2.4 : Breadboard

Jumper Wires: A jumper wire is an electric wire that connects remote electric circuits used for printed circuit boards.

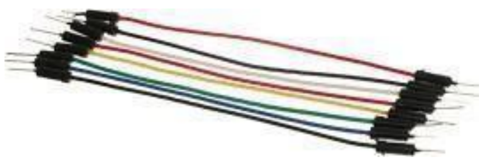


Fig 2.5 : Jumper Wires

TINKERCAD DIAGRAM:

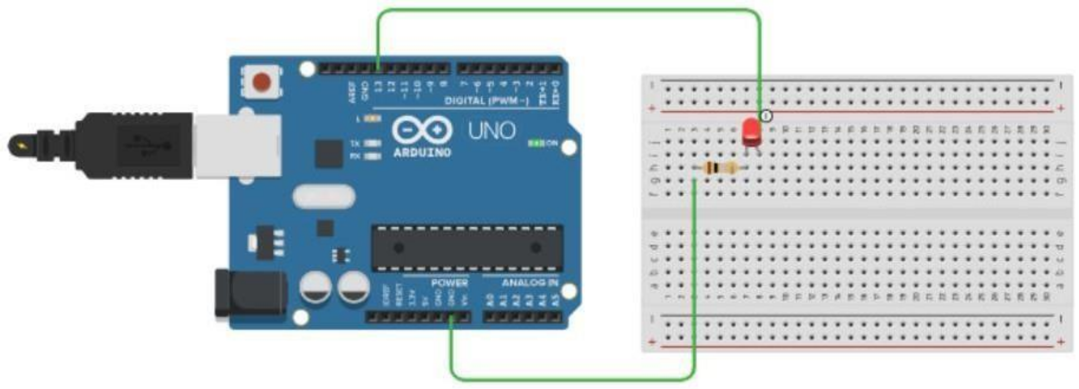


Fig 2.6 : Blink a single LED using Arduino and breadboard

CODE:

```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(500);
  digitalWrite(13, LOW);
  delay(500);
}
```

RESULTS:

In this experiment, we learnt how to blink an LED using Arduino UNO.

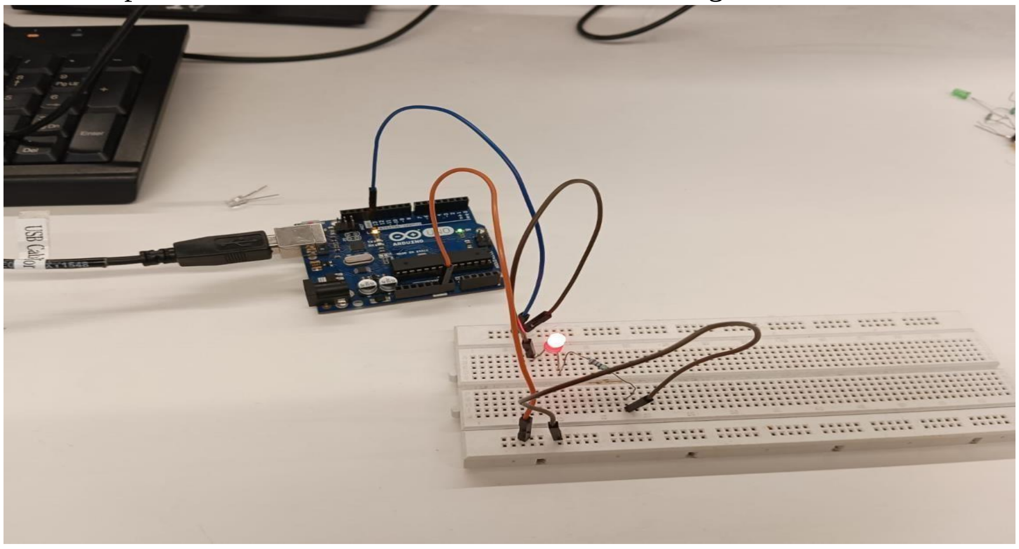


Fig 2.7 : Blink a single LED using Arduino and breadboard

Signature of faculty member

EXPERIMENT-3

OBJECTIVE: To blink multiple LEDs using Arduino Uno and breadboard.

SOFTWARE USED: Arduino

HARDWARE USED:

Sr No.	Name of the Component	Value
1.	Arduino Uno Board	1
2.	Breadboard	1
3.	Jumper Wires	9
4.	LED	5
5.	Resistor	220 ohm

Table 3.1: Hardware Used

THEORY:

Resistor: Resistors are used in virtually all electronic circuits and many electrical ones. Resistors, as their name indicates resist the flow of electricity and this function is key to the operation most circuits.

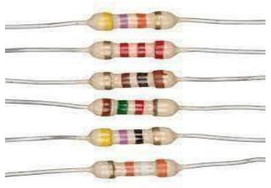


Fig 3.1: Various Resistors

LED: A light-emitting diode (LED) is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons (Energy packets).

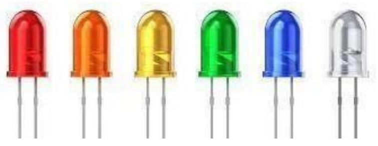


Fig 3.2: LEDs

Arduino Uno Board: The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc.

Fig 3.3: Arduino Uno R3

Breadboard: A breadboard is used to place components (resistor, capacitor, LED's etc.) that are wired together. It is used to make temporary circuits.

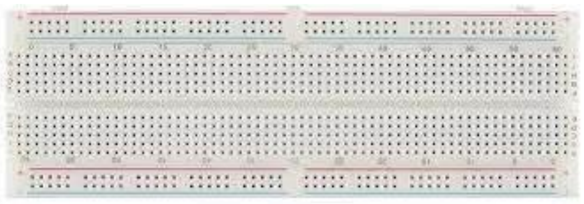


Fig 3.4: Breadboard

Jumper Wires: A jumper wire is an electric wire that connects remote electric circuits used for printed circuit boards.

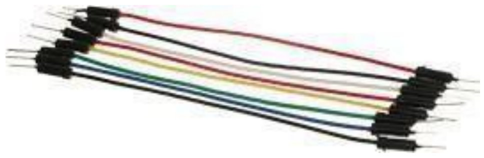


Fig 3.5: Jumper wires

TINKERCAD DIAGRAM:

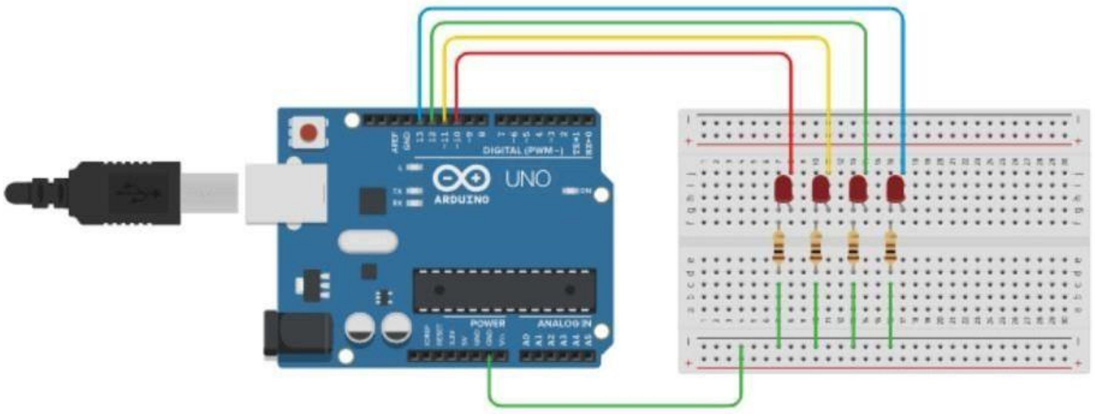


Fig 3.6: Circuit for 5LEDs blinking in forward and reverse pattern

CODE :

```
void setup(){
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
}

void loop(){
  for(int i=9; i<14; i++){
    digitalWrite(i, HIGH);
      delay(500);
      digitalWrite(i, LOW);
      delay(500);
  }
}
```

RESULTS:
Hence, we successfully blinked multiple LEDs using the above components by making suitable connections and coded using for loop.

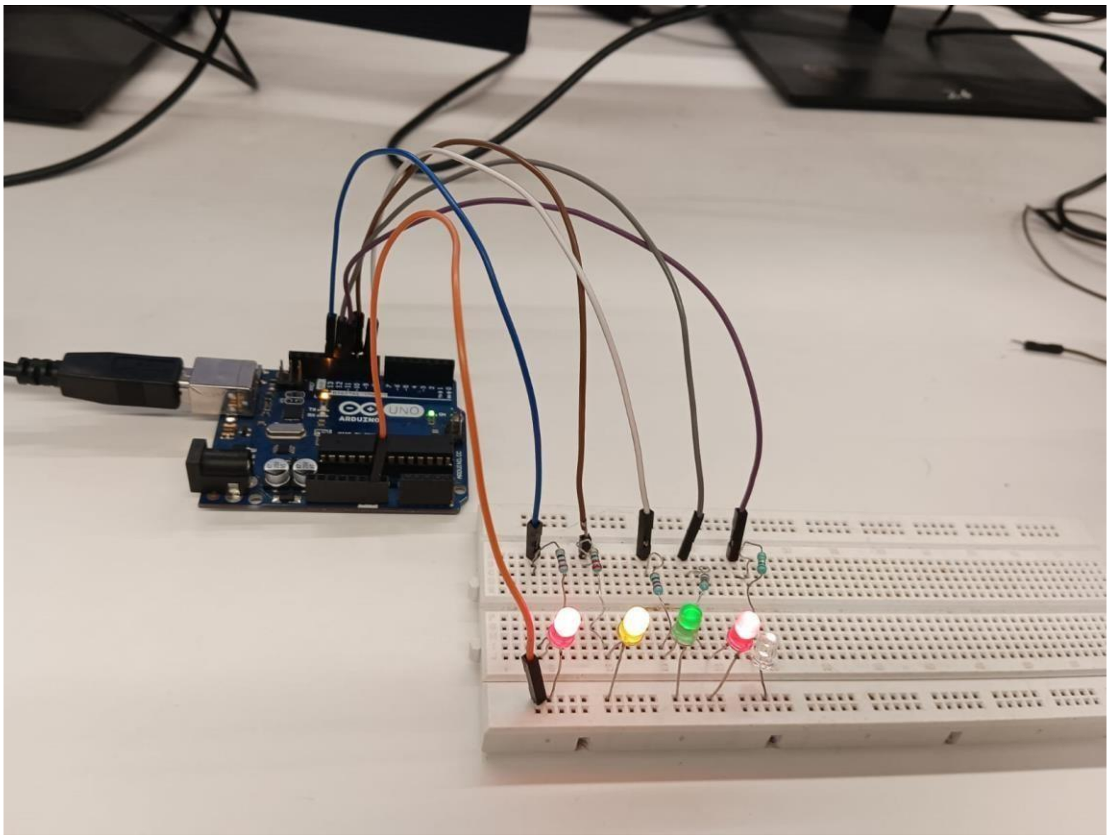


Fig 3.7: Synchronised blinking of LEDs

Signature of faculty member

EXPERIMENT-4

OBJECTIVE: Write a program to design a pattern of sequence of multiple LED’s using for loop in Arduino.

SOFTWARE USED: Arduino

HARDWARE USED:

Sr No.	Name of the Component	Value
1.	Arduino Uno Board	1
2.	Breadboard	1
3.	Jumper Wires	9
4.	LED	5
5.	Resistor	220 ohm

Table 4.1: Hardware Used

THEORY:

In this experiment we use the concept of array,for ,loops and conditonal statements of arduino programming along with digitalWrite() function to design a pattern of sequence of multiple LED’s

ARRAY: An array is a collection of variables that are accessed with an index number. Arrays in the C++ programming language Arduino sketches are written in can be complicated, but using simple arrays is relatively straightforward. Arrays are zero indexed, that is, referring to the array initialization above, the first element of the array is at index 0.

FOR LOOP: The for statement is used to repeat a block of statements enclosed in curly braces. An increment counter is usually used to increment and terminate the loop. The for statement is useful for any repetitive operation, and is often used in combination with arrays to operate on collections of data/pins.

CONDITIONAL STATEMENT: The if() statement is the most basic of all programming control structures. It allows you to make something happen or not, depending on whether a given condition is true or not.

DigitalWrite(): Write a HIGH or a LOW value to a digital pin. If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW.If the pin is configured as an INPUT, digitalWrite() will enable (HIGH) or disable (LOW) the internal pullup on the input pin. It is recommended to set the pinMode() to INPUT_PULLUP to enable the internal pull-up resistor. .If you do not set the pinMode() to OUTPUT, and connect an LED to a pin, when calling digitalWrite(HIGH), the LED may appear dim. Without explicitly setting pinMode(), digitalWrite() will have enabled the internal pull-up resistor, which acts like a large current-limiting resistor.

DELAY(): The delay() function allows you to pause the application of your arduino program for a specific period.

TINKERCAD DIAGRAM:

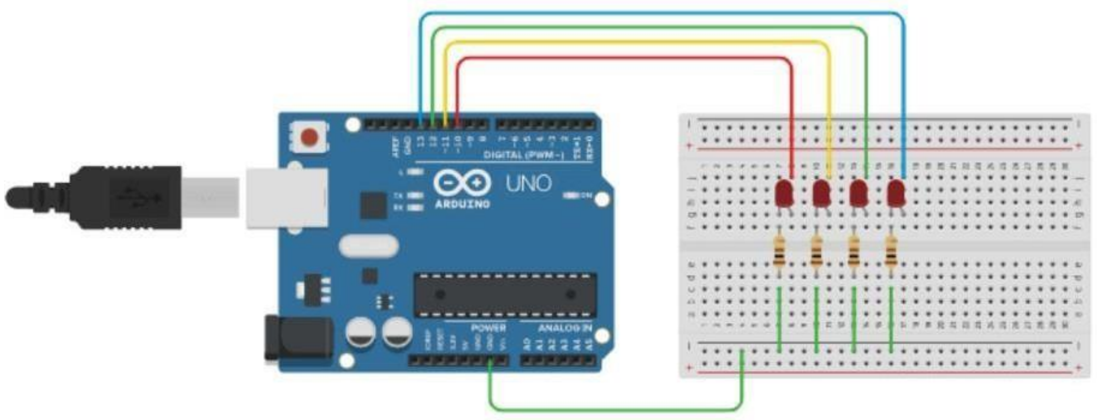


Fig 4.1: Sequence of multiple LED’s

CODE 1:

```
//Using for Loop
void setup(){
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
}

void loop(){
  for(int i=9; i<14; i++)
  {
    digitalWrite(i, HIGH);
      delay(500);
      digitalWrite(i, LOW);
      delay(500);
  }
}
```

CODE 2:

```
//Using while Loop
void setup(){
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
}

void loop(){
  for(int i=9; i<14; i++){
    digitalWrite(i, HIGH);
      delay(500);
      digitalWrite(i, LOW);
      delay(500);
  }
}
```

RESULTS:
In this experiment we made a pattern where the LEDs which were placed at odd positions lit up first then the LEDs which were placed at even positions lit up. They were dimmed in the reverse order.

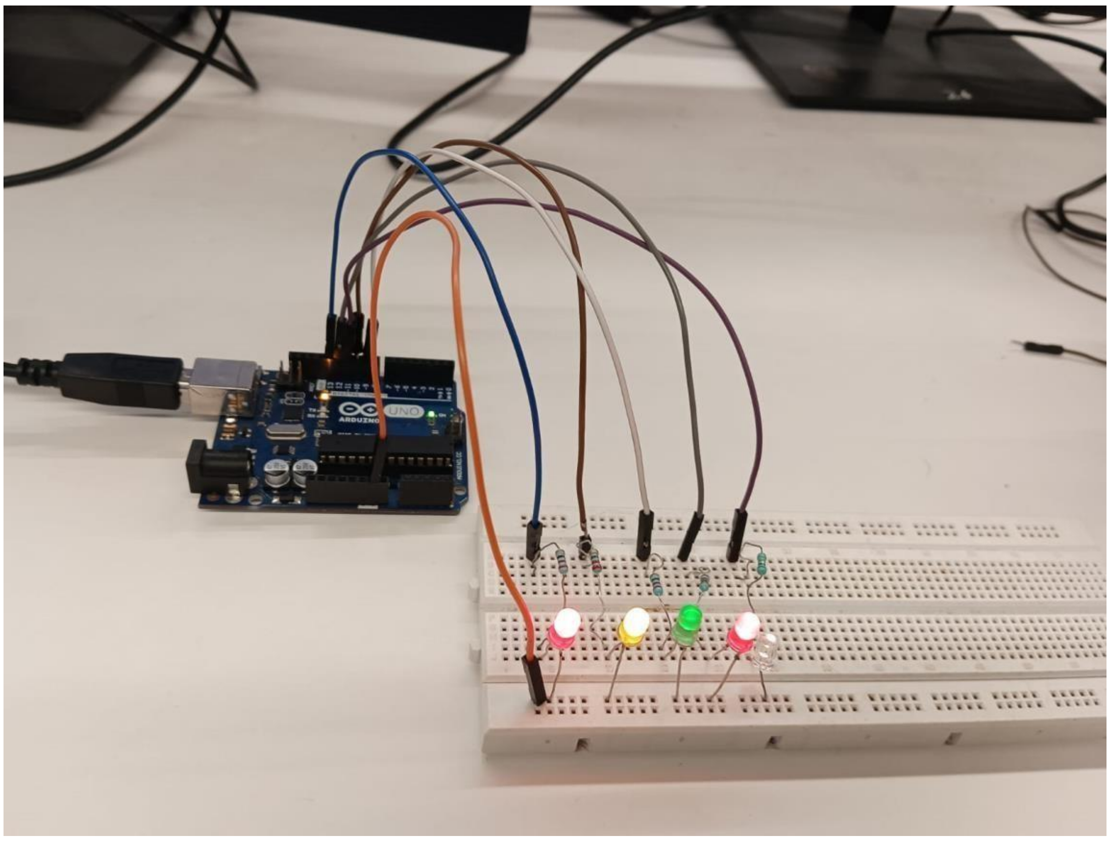


Fig 4.2: Sequence of multiple LED's

Signature of faculty member

EXPERIMENT-5

OBJECTIVE: Write a program to demonstrate sending data from the computer to the Arduino board and control brightness of LED.

SOFTWARE USED: Arduino

HARDWARE USED:

Sr No.	Name of the Component	Value
1.	Arduino Uno Board	1
2.	Breadboard	1
3.	Jumper Wires	3
4.	LED	1
5.	Resistor	220 ohm

Table 5.1 : Hardware Used

THEORY:
In this experiment we use the concept of array,for loops and conditonal statements of arduino programming along with digitalWrite() AND analogWrite() function to demonstrate sending data from the computer to the Arduino board and control brightness of LED.

ARRAY: An array is a collection of variables that are accessed with an index number. Arrays in the C++ programming language Arduino sketches are written in can be complicated, but using simple arrays is relatively straightforward**. Arrays are zero** indexed, that is, referring to the array initialization above, the first element of the array is at index 0.

SIZEOF(): It returns the size of the given parameter in bytes.

FOR LOOP: The for statement is used to repeat a block of statements enclosed in curly braces. An increment counter is usually used to increment and terminate the loop. The for statement is useful for any repetitive operation, and is often used in combination with arrays to operate on collections of data/pins.

CONDITIONAL STATEMENT: The if() statement is the most basic of all programming control structures. It allows you to make something happen or not, depending on whether a given condition is true or not.

DigitalWrite(): Write a HIGH or a LOW value to a digital pin. If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW.If the pin is configured as an INPUT, digitalWrite() will enable (HIGH) or disable (LOW) the internal pullup on the input pin. It is recommended to set the pinMode() to INPUT_PULLUP to enable the internal pull-up resistor. .If you do not set the pinMode() to OUTPUT, and connect an LED to a pin, when calling digitalWrite(HIGH), the LED may appear dim. Without explicitly setting pinMode(), digitalWrite() will have enabled the internal pull-up resistor, which acts like a large current-limiting resistor.

DELAY(): The delay() function allows you to pause the application of your arduino program for a specific period.

analogWrite(): Writes an analog value (PWM wave) to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds. After a call to analogWrite(), the pin will generate a steady rectangular wave of the specified duty cycle until the next call to analogWrite() on the same pin.

TINKERCAD DIAGRAM:

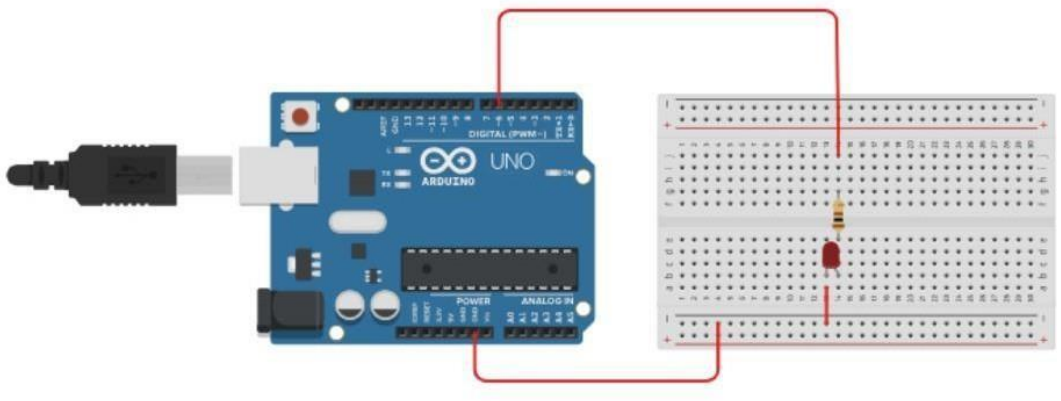


Fig 5.1: Control brightness of LED

CODE:

```
int a;
void setup ()
{
  Serial.begin(9600);
  pinMode(6, OUTPUT);
}
void loop ()
{
  Serial.println("Enter Intensity:  ");
  a= Serial.parseInt(); analogWrite(6,a);
  delay(500);
  analogWrite(6,0);
  delay(500);
}
```

RESULTS:

In this experiment we successfully learned how to control the brightness of LED using ‘analogWrite’ function. We also got to know that the brightness can range between 0 (always off) and 255 (always on). If we set the value above 255 then the result we get is Value%255.

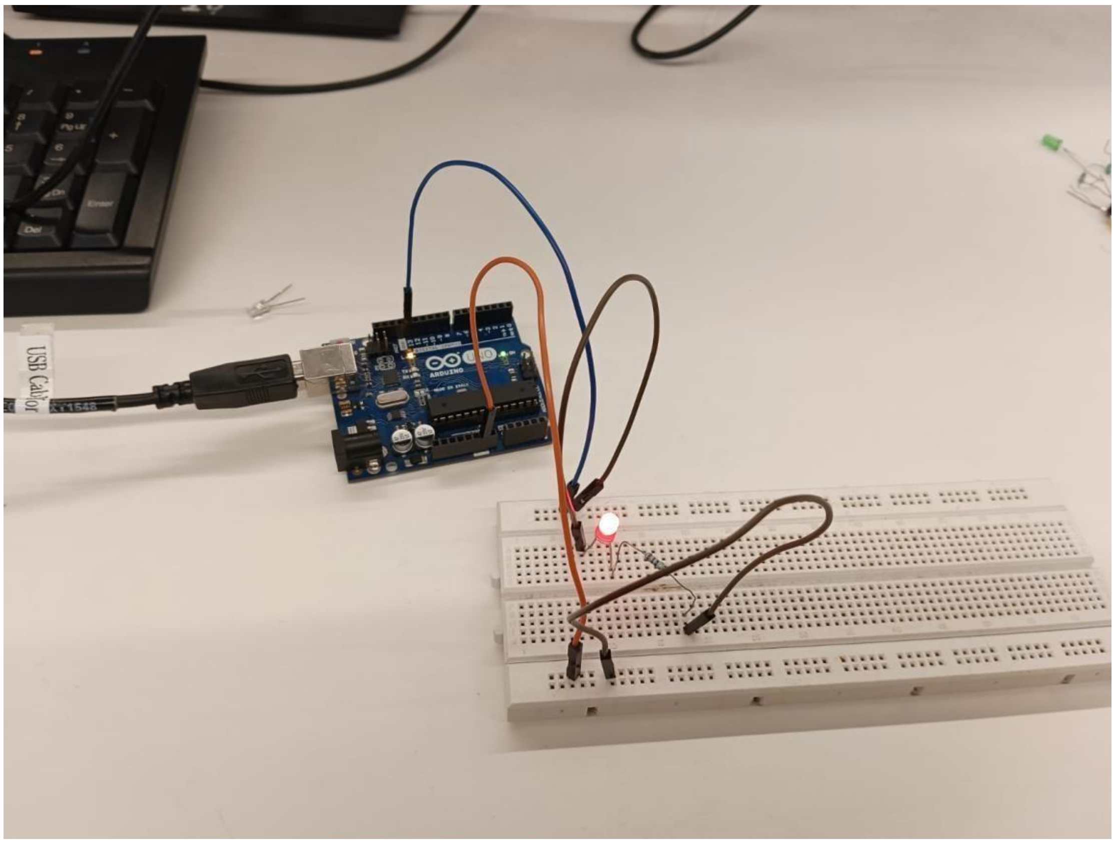


Fig 5.2: Control brightness of LED

Signature of faculty member

EXPERIMENT-6

OBJECTIVE: *Serial Communication:*
WAP to.
****
Roll_No.____
****
Name:____
****
Branch:_______________________________________
****

SOFTWARE USED: Arduino IDE

HARDWARE USED:

Sr No.	**Name of the Component**	**Value**
1.	Arduino Uno Board	1

Table 6.1 : Hardware Used

THEORY:
Serial Communication: Serial is used for communication between the Arduino board and a computer or other devices. Serial data transfer is when we transfer data one bit at a time, one right after the other. When you upload the data to the Arduino, the bits are shoved out one at a time through the USB cable to the Arduino where they are stored in the main chip.

1. Serial.print();
The serial.print ( ) in Arduino prints the data to the serial port. The printed data will be visible in the serial monitor.

2. Serial.read();
Serial.read() is a function of the Serial library. What it does is read out the first available byte
from the serial receive buffer. Say you had sent the phrase "Sub Sandwich" to your Arduino.
This means you had put 12 bytes into your serial receive buffer.

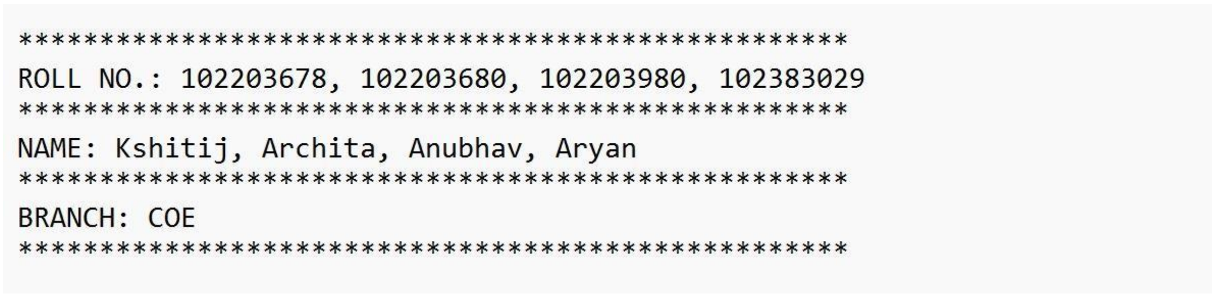
3. Serial.write();
Writes binary data to the serial port. This data is sent as a byte or series of bytes. Let's say you need to send the number 217. The binary (1's and 0's) representation of this number is 11011001. Using the command Serial.write(217) will literally just send 11011001 across the line.

CODE:

```
void setup()
{
  Serial.begin(9600);

  for(int i=0; i<50; i++)
  {
    Serial.print("*");
  }
  Serial.println(" ");

  Serial.println("ROLL   NO.: 102203678,   102203680,   102203980,
102383029");

  for(int i=0; i<50; i++)
  {
    Serial.print("*");
  }
  Serial.println(" ");

  Serial.println("NAME:   Kshitij, Archita,  Anubhav,  Aryan");

  for(int i=0; i<50; i++)
  {
    Serial.print("*");
  }
  Serial.println(" ");

  Serial.println("BRANCH: COE");
   for(int i=0; i<50; i++)
  {
    Serial.print("*");
  }
}

void loop()
{
  //sample code
}
```

RESULT:

We used the ‘serial.println’ function and for loops to print the pattern to the serial monitor



```
**************************************************
ROLL NO.: 102203678, 102203680, 102203980, 102383029
**************************************************
NAME: Kshitij, Archita, Anubhav, Aryan
**************************************************
BRANCH: COE
**************************************************
```

Fig 6.1: Printed pattern using for loop

Signature of faculty member

EXPERIMENT-7

OBJECTIVE: WAP to show dimmer effect where LED 1 should display values between 1-50
LED 2 = 51-100
LED 3 = 101-150
LED 4 = 151-200
LED 5 = 201-255

SOFTWARE USED: Tinkercad Simulator

HARDWARE USED:

Sr No.	Name of the Component	Value
1.	Arduino Uno Board	1
2.	Breadboard	1
3.	Jumper Wires	11
4.	LED	5
5.	Resistor	220 ohm

Table 7.1 : Hardware Used

THEORY:

In this experiment we use the concept of array,for loops and conditonal statements of arduino programming along with digitalWrite() AND analogWrite() function to show dimmer effect where different LED will display different values between 1-255.

ARRAY: An array is a collection of variables that are accessed with an index number. Arrays in the C++ programming language Arduino sketches are written in can be complicated, but using simple arrays is relatively straightforward. Arrays are zero indexed, that is, referring to the array initialization above, the first element of the array is at index 0.

SIZEOF(): It returns the size of the given parameter in bytes.

FOR LOOP: The for statement is used to repeat a block of statements enclosed in curly braces. An increment counter is usually used to increment and terminate the loop. The for statement is useful for any repetitive operation, and is often used in combination with arrays to operate on collections of data/pins.

CONDITIONAL STATEMENT: The if() statement is the most basic of all programming control structures. It allows you to make something happen or not, depending on whether a given condition is true or not.

DigitalWrite(): Write a HIGH or a LOW value to a digital pin.If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW. If the pin is configured as an INPUT, digitalWrite() will enable (HIGH) or disable (LOW) the internal pullup on the input pin. It is recommended to set the pinMode() to **INPUT_PULLUP** to enable the internal pull-up resistor. .If you do not set the **pinMode()** to **OUTPUT**, and connect an LED to a pin, when calling **digitalWrite(HIGH)**, the LED may appear dim. Without explicitly setting **pinMode()**, **digitalWrite()** will have enabled the internal pull-up resistor, which acts like a large current-limiting resistor.

DELAY: The delay() function allows you to pause the application of your arduino program fro a specific period.

analogWrite(): Writes an analog value (PWM wave) to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds. After a call to analogWrite(), the pin will generate a steady rectangular wave of the specified duty cycle until the next call to analogWrite() on the same pin. Arduino Uno R3 has 6 PWM pins that are **3, 5, 6, 9, 10, and 11**. These pins are marked with the negation sign "~". These pins can generate a pulse as per the given inputs. Arduino supports an 8-bit wide pulse that can have 256 possible levels (0 to 255).
PULSE WIDTH MODULATION (PWM):Pulse Width Modulation is a technique to get variable voltage in terms of Digital Input. PWM device generates ON and OFF pulses according to the control signal, which determines the desired voltage level.
PWM is used to control the amount of power delivered to the load. It is commonly used for controlling the brightness of LED, the speed of motors, etc.

TINKERCAD DIAGRAM:

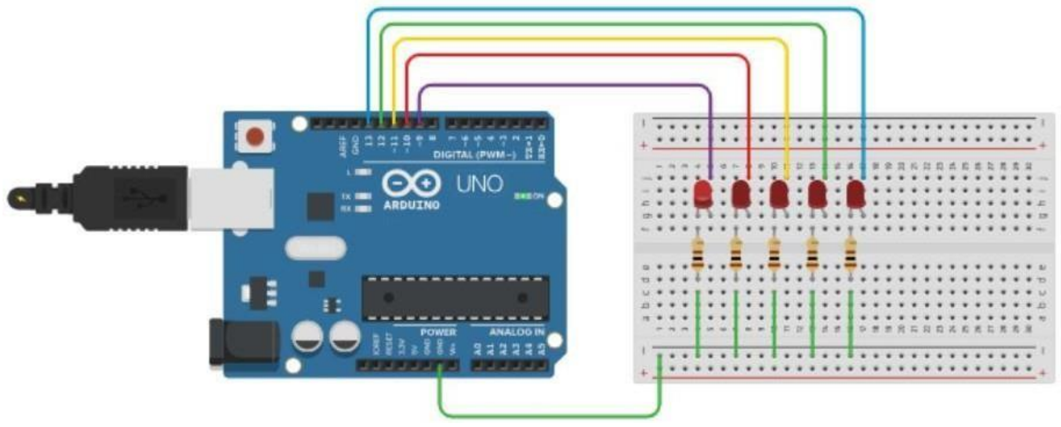


Fig 7.1 Circuit for Change intensity of multiple LEDs

CODE:

a. Using Digital Write

```
1. void setup()
    {
     pinMode(3, OUTPUT);
    }

    void loop() {

      for (int i = 10; i <= 1000; i=i+10)
      {
    digitalWrite(3,
    HIGH);
                    delay(
    50);
    digitalWrite(3,
    LOW);

            delay(50);
      }
    }
2. int

    seq[]={9,10,11,12,1
    3};
void setup() {

      pinMode(9,
    OUTPUT);
    pinMode(10,
    OUTPUT);
     pinMode(11,
    OUTPUT);
     pinMode(12,
    OUTPUT);
      pinMode(13, OUTPUT);
    }

    void loop() {


      for(int j=0;j<5;j++)
      {
            for (int i = 10; i <= 1000; i=i+10)
            {
    digitalWrite(seq[j],
    HIGH);
    delay(50);
    digitalWrite(seq[j],
LOW);
delay(50);
```

```
            }
  }
}
```

b. Using Analog Write

```
1. void setup()
{
  pinMode(3,
OUTPUT);
}

void loop()
{
  for(int
i=50;i<255;i++)
  {
    analogWrite(3, i);
            delay(50);
  }
}
2. void setup()
{
pinMode(9,
OUTPUT);
pinMode(10,
OUTPUT);
pinMode(11,
OUTPUT);
pinMode(12,
OUTPUT);
pinMode(13,
OUTPUT);
}

void loop()
{
  analogWrite(9,50);
delay(500);
analogWrite(10,100);

delay(500);
 analogWrite(11,150);

delay(500);
analogWrite(12,200);

delay(500);
analogWrite(13,250);

delay(500); }
```

RESULT:

We executed dimmer effect by using for loops for each LED and then used ‘analogWrite’ function to manage the brightness according to the question.

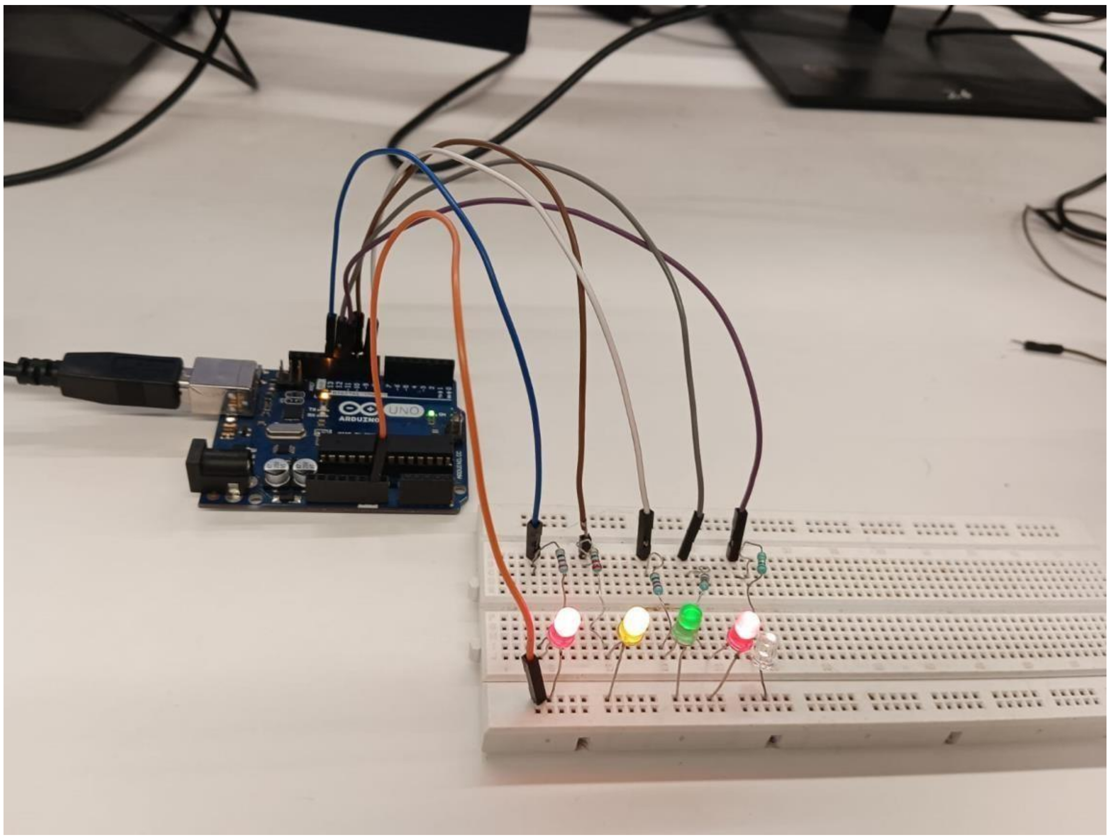


Fig 7.2 Circuit for Change intensity of multiple LEDs

Signature of faculty member

EXPERIMENT-8

OBJECTIVE: Write a program to change the intensity of the given LED's for the sequence 35214 in for both forward and reverse order.

SOFTWARE USED: Arduino

HARDWARE USED:

Sr No.	Name of the Component	Value
1.	Arduino Uno Board	1
2.	Breadboard	1
3.	Jumper Wires	11
4.	LED	5
5.	Resistor	220 ohm

Table 8.1: Hardware Used

THEORY:

In this experiment, we initialise different LED in the setup() function. Setup() is usedto initialize variables, pin modes, start using libraries, etc. The setup() function will only run once, after each powerup or reset of the Arduino board.We take pins 3,5,6,9,10 as the output in pinmode() function as these are the PWM pins. And in the loop() function using analogWrite() function change the intensity of the LEDs in the sequence 35214 for both forward and reverse order.

Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between the full Vcc of the board (e.g., 5 Von UNO, 3.3 V on a MKR board) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off. The duration of "on time" is called the pulse width. To get varying analog values, you change, or modulate, that pulse width. If you repeat this on-off pattern fast enough with an LED for example, the result is as if the signal is a steady voltage between 0 and VCC controlling the brightness of the LED.

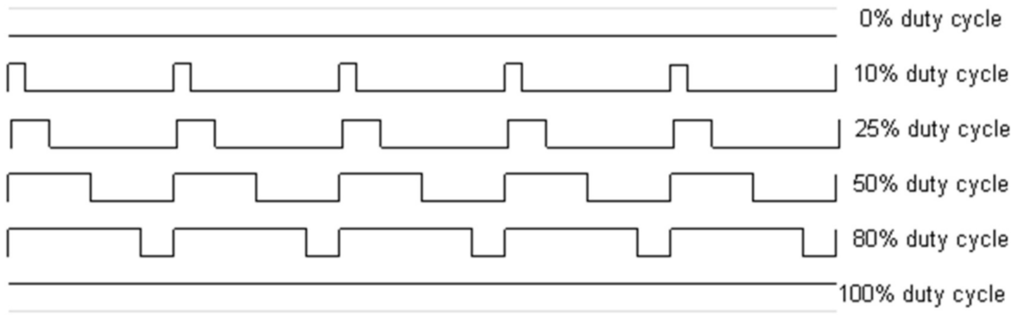


Fig 8.1 PWM Signal

The Arduino's programming language makes PWM easy to use; simply call analogWrite(pin, dutyCycle), where dutyCycle is a value from 0 to 255, and pin isone of the PWM pins (3, 5, 6, 9, 10, or 11). The analogWrite() function provides a simple interface to the hardware PWM, but doesn't provide any control over frequency.

TINKERCAD DIAGRAM:

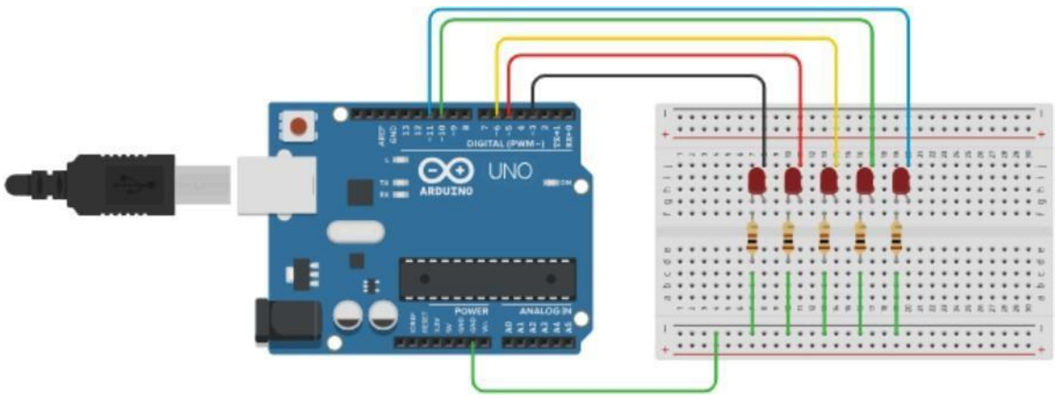


Fig 8.2 Change the intensity of the given LED's for the sequence

CODE:

```
void setup()
        {
        pinMode   (3,  OUTPUT);
        pinMode   (5,  OUTPUT);
        pinMode   (6,  OUTPUT);
        pinMode (10, OUTPUT);
        pinMode (11, OUTPUT);
                }
void loop ()
{
  for(int i=1; i<255; i++){
if(i<50){
analogWrite(6,i);
delay(10);
```

```
}
if(i>51 && i<100){
analogWrite(10,i);
delay(10);
}
if(i>101 && i<150){
analogWrite(5,i);
delay(10);
}
if(i>151 && i<200){
analogWrite(3,i);
delay(10);
}
if(i>201 && i<255){
analogWrite(9,i);
delay(10);
}
}
for(int i=1; i<255; i++){
if(i<50){
analogWrite(9,i);
delay(10);
}
if(i>51 && i<100){
analogWrite(3,i);
delay(10);
}
if(i>101 && i<150){
```

```
analogWrite(5,i);

delay(10);

}

if(i>151 && i<200){

analogWrite(10,i);

delay(10);

}

if(i>201 && i<255){

analogWrite(6,i);

delay(10);

}

    }


    }
```

RESULT:

this experiment, we learnt how In to change the intensity of the given LED's for the sequence 35214 in for both forward and reverse order.

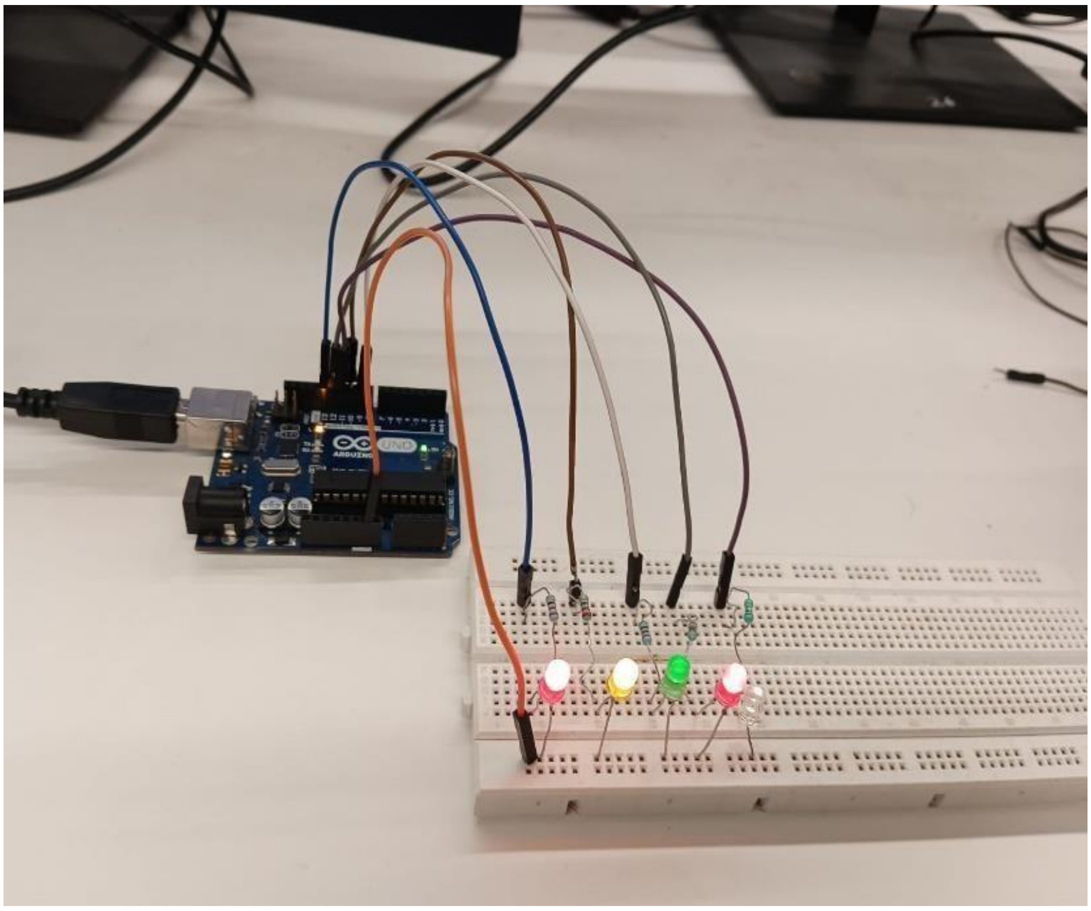


Fig 8.3 Change the intensity of the given LED’s for the sequence

Signature of faculty member

EXPERIMENT-9

OBJECTIVE: Write a program to demonstrate control of DC Motor using forward, backward, left, right turn motion and clock- wise/anti clock- wise rotation.

SOFTWARE USED: Arduino IDE.

HARDWARE USED:

Sr No.	Name of the Component	Value
1.	Nvis 3302ARD RoboCar	1
2.	Arduino UNO	1
3.	USB Cable	1

Table 9.1: Hardware Used

THEORY:

Nvis 3302ARD is capable of sensing environment using various sensor modules and acts accordingly. Nvis RoboCar is a ready assembled unit consisting of strong chassis wheels with different Sensor modules mounted on it. The machine is driven by DC motors which are powered by rechargeable batteries. This Nvis 3302ARD is Atmega328P Micro-controller RoboCar. We can design user defined functions in the Arduino IDE to make the buggy move in our own specified directions like left, right, forward, backward, clockwise, and anti- clockwise by setting the pins 5, 6, 7 ,8 on Nvis 3302ARD RoboCar either HIGH/LOW.

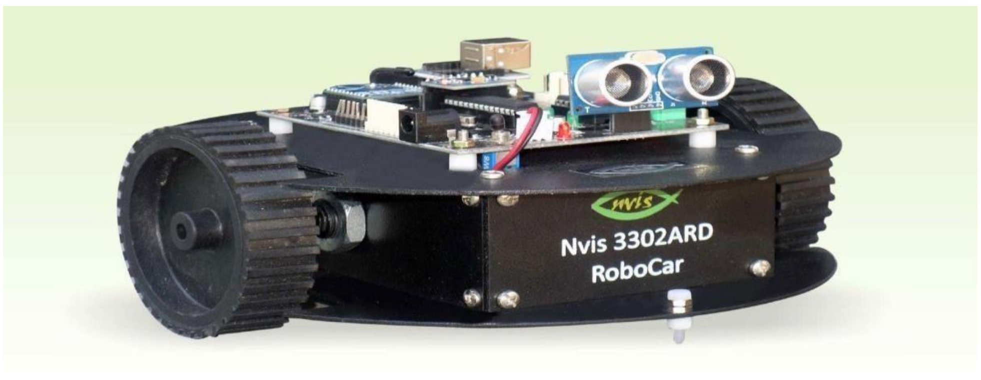


Figure 9.1 Nvis 3302ARD Model

CODE:

```
void setup()
{
  pinMode(5,OUTPUT);
  pinMode(6,OUTPUT);
  pinMode(7,OUTPUT);
  pinMode(8,OUTPUT);
  Serial.begin(9000);
}
void forward()
{
  digitalWrite(5,HIGH);
  digitalWrite(6,LOW);
  digitalWrite(7,LOW);
  digitalWrite(8,HIGH);
}
void back()
{
  digitalWrite(5,LOW);
  digitalWrite(6,HIGH);
  digitalWrite(7,HIGH);
  digitalWrite(8,LOW);
}
void left()
{
  digitalWrite(5,HIGH);
  digitalWrite(6,LOW);
  digitalWrite(7,LOW);
  digitalWrite(8,LOW);
}
void right()
{
  digitalWrite(5,LOW);
  digitalWrite(6,LOW);
  digitalWrite(7,LOW);
  digitalWrite(8,HIGH);
}
void CW()
{
  digitalWrite(5,LOW);
  digitalWrite(6,HIGH);
  digitalWrite(7,LOW);
  digitalWrite(8,HIGH);
}
void ACW()
{
  digitalWrite(5,HIGH);
  digitalWrite(6,LOW);
  digitalWrite(7,HIGH);
  digitalWrite(8,LOW);
}
void stop()
{
   digitalWrite(5,LOW);
  digitalWrite(6,LOW);
  digitalWrite(7,LOW);
  digitalWrite(8,LOW);

}
void loop()
{
  forward();
  delay(3000);
  back();
  delay(3000);
  left();
  delay(3000);
  right();
  delay(3000);
  CW();
  delay(3000);
  ACW();
  delay(3000);
  stop();
  delay(1000000);
}
```

RESULTS:
Hence, we successfully performed the given movements of Robo car (Buggy) using functions from the above components by making suitable connections and code.We saw that the buggy can easily move in forward, then backward, then left, then right, then clockwise, then anticlockwise direction and finally stop in the end after a gap of 2 seconds each.

Signature of faculty member